95-427 [WGM 1897] Patent

UNITED STATES UTILITY PATENT APPLICATION

FOR

GENERIC COMMAND INTERFACE FOR MULTIPLE EXECUTABLE ROUTINES

INVENTORS:

Jeffrey Wheeler of Glen Allen, Virginia
Paul Mustoe of Richmond, Virginia

PREPARED BY:

Leon R. Turkevich, Esq.
2000 M STREET, N.W., 7th Floor
WASHINGTON, D.C. 20036-3307
(202) 261-1059

GENERIC COMMAND INTERFACE FOR MULTIPLE EXECUTABLE ROUTINES

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

The present invention relates to command and interface control of Operating Administration and Monitoring (OAM) executable routines within software systems.

DESCRIPTION OF THE RELATED ART

Operating Administration and Monitoring (OAM) tools are software-based resources used as administration and/or diagnostic tools for complex processor- based executable software systems, such as software-based unified messaging software systems. A subset of OAM tools includes Real Time Monitoring (RTM) programs, used to monitor and control selected states and processes within the software based system. For example, a given RTM program may generate a real-time display (i.e., "a screen") of selected parameters during execution of a prescribed process; the RTM program may also provide a diagnostic resource that enables resetting of various states or variables within the prescribed process. Other administration and diagnostic tools include external binary files that execute in response to a procedure call, and Simple Network Management Protocol (SNMP) agents or scripts configured for generating an e-mail message as an alarm in response to a detected event.

Hence, system administrators may attempt to utilize multiple tools within a software system in order to increase the available administration and diagnostic tools for improved system performance. The use of multiple RTM programs and other OAM tools, however, requires the users to remember the names and syntaxes of numerous commands for the respective RTM programs and OAM tools. Hence, an increase in the number of OAM tools would result in the system administrator needing to develop expertise in the command names and syntaxes for the respective OAM tools.

SUMMARY OF THE INVENTION

There is a need for an arrangement that integrates multiple RTM programs and command and control functionality for a user, without the necessity of learning the respective command formats

and syntax.

There is also a need for arrangement that enables a simple command language to be utilized for control of multiple RTM programs having respective command formats.

These and other needs are attained by the present invention, where a processor based system having a parser is configured for validating a generic command received from a user relative to a command parse tree. The command parse tree includes multiple elements, each specifying at least one corresponding generic command component and a corresponding at least one command action value. The parser, upon identifying a best match among the elements, issues a prescribed command for a selected one of the management programs according to the corresponding command format based on the selected command action value. Hence, a user may control multiple management programs having respective command formats, by using a set of generic commands that are independent from the command formats, eliminating the necessity that the user needs to learn the detailed command formats and syntax.

One aspect of the present invention provides a method in a processor-based system configured for executing a plurality of management programs according to respective command formats. The method includes receiving a generic command from the user, and validating the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format, the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the validating step including identifying one of the elements as a best match relative to the generic command. The method also includes issuing a prescribed command of a selected one of the management programs according to the corresponding command format, based on the identified one element.

Another aspect of the present invention provides a system configured for executing a plurality of management programs according to respective command formats. The system includes a parser having a command parse tree configured for validating a generic command received from a user, the command parse tree configured for specifying valid generic commands relative to a prescribed generic command format and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the parser

identifying one of the elements as a best match relative to the generic command. The system also includes a plurality of translators configured for issuing commands for the management programs according to respective command formats, the parser outputting a prescribed command to a selected one of the translators based on the identified one element.

Additional advantages and novel features of the invention will be set forth in part in the description which follows and in part will become apparent to those skilled in the art upon examination of the following or may be learned by practice of the invention. The advantages of the present invention may be realized and attained by means of instrumentalities and combinations particularly pointed out in the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

Reference is made to the attached drawings, wherein elements having the same reference numeral designations represent like elements throughout and wherein:

Figure 1 is a diagram of a system configured for executing multiple management programs according to respective command formats based on a generic command set according to an embodiment of the present invention.

Figure 2 is a diagram illustrating in detail the parser of Figure 1 according to an embodiment of the present invention.

Figures 3A and 3B are diagrams illustrating the validation of generic commands by the parser of Figure 1 according to an embodiment of the present invention.

BEST MODE FOR CARRYING OUT THE INVENTION

Figure 1 is a diagram of a system configured for executing a plurality of management programs according to respective command formats according to an embodiment of the present invention. The processor based system 10 includes a user input interface 12, for example a terminal interface, that enables a user to input a generic command string, described below. The processor based system 10 also includes a parser 14 configured for validating the generic command received by the user input interface 12 from the user, and translators 16 configured for issuing commands to respective management programs 18 according to respective command formats. As shown in Figure

1, the management programs 18, implemented for example by different OAM tools such as RTM programs, may be executed within the processor based system or externally as external agents accessible using a prescribed application programming interface (API). The management programs 18 may provide different administration and maintenance functions, for example initiating various real-time screens used to monitor the internal state of executable processes within the software based system 10; alternately, different tools 18 may allow the user to control the various states within the various component of the software based system 10 via external programs (e.g., programs 18c or 18d), or may be used to issue external alarms (e.g., SNMP manager scripts) for external routines such as message waiting indicator routines.

A disadvantage of utilizing many different tools 18 is that each tool 18 tends to have its own screen and/or command, providing difficulties for the system administrator to determine which tool is the best tool (and/or which is the best syntax) to use for a given problem.

According to the disclosed embodiment, the parser 14 and the translators 16 provide a unified administration and diagnostic tool which incorporates the functionality of all external administrative executable binary files, RTM programs, agent manipulation scripts, and various requested snapshot queries, as well as including an extensive help system. In particular, the parser 14 and the translators 16 provide a generic command syntax that integrates the functionality of the different tools 18 and that automatically selects the appropriate command for the best tool for executing a given generic command. As illustrated in Part A. of the attached appendix, the new syntax provides a generic instruction set that provides an abstraction of the tool-specific command formats and syntax, enabling a user to issue command based on the relative functions, as opposed to the specific syntax for a corresponding tool 18.

Figure 2 is a diagram illustrating in detail the parser 14 of Figure 1 according to an embodiment of the present invention. The parser 14 includes a command word translation table 20 and a command parse tree 22. The command word translation table 20 is configured for storing, for each prescribed command word 26, a corresponding token value 28 that is used by the parser 14 to identify a specific command for a selected one of the translators 16. In particular, the command word translation table 20 includes all the command words 26 that are valid according to the generic syntax, illustrated for example in Part B of the attached appendix.

The parser 14 is configured for validating a received generic command by comparing each input command word to the command parse tree 22 to determine for the received generic command a tree element 24 identified as a best match. Each tree element 24 includes at least one token-command key pair 30 that specifies a token (T) 28 and a corresponding command key (CK) 32, enabling the parser 14 to identify the appropriate prescribed command based on the command key specified for the matching token. In particular, the parser 14 recursively traverses the command parse tree 22 for each command word to identify the best match for the generic command. If only a portion of the generic command is identified as valid (e.g., only the first three command words are valid), the parser 14 selects the command key 32 for the matching token 28 from the last valid tree element 24.

Figure 3 is a diagram illustrating the method of validating a received generic command and translating the received generic command into a command for a specific management program according to an embodiment of the present invention. The operations described with respect to Figures 2 and 3 can be implemented as executable code that is stored on a computer readable medium (e.g., a hard disk drive, a floppy drive, a random access memory, a read only memory, an EPROM, a compact disk, etc). The method begins in step 40, wherein the parser begins parsing the first word of the received generic command by comparing the first input command word to the command word translation table 20 for identification of a matching token 28. For example, assume that the parser 14 receives the valid command "watch tcp connections". The parser identifies the token value "8" as corresponding to the first command word "watch". The parser 14 than traverses the command parse tree 22 in step 42 to search for the matching token 28. As illustrated in Figure 2, the parser 14 locates the matching token in the first tree element 24a. If the parser 14 determines in step 44 that the first command word is valid, the parser 14 continues searching the next command word in step 46. If the first command word is invalid based on no match in the first element 24a of the command parse tree, the parser 14 returns an invalid command message to the user in step 56.

The parser 14 then parses the next word (e.g., "tcp") of the received generic command in step 46 by locating the corresponding token 28 (e.g., "6" for "tcp") in the table 20, and then traversing in step 48 the tree elements that depend from the matched tree element 24a (e.g., 24b). The parser 14 determines a match between the token 28 ("6") corresponding to the command word "tcp" in the

token-command key pair 30d in step 50, enabling the parser to continue for the next command word. As described above, the parser 14 repeats the process in step 52 for the third command word "connections" having the token "2" and identifying a match between the entire generic command and the token-command key 30 specified in the tree element 24c. The parser 14 identifies in step 54 the prescribed command for a selected one of the translators 16 based on the value of the command key 32 within the matching token-command key pair 30 (e.g., "CK=3") of the last valid command word, which maps to a translation table that specifies a specific command for a specific translator 16.

As described above, the parser 14 can identify a command key 32 even if only a portion of the command is valid. Assume for example that the parser 14 receives the invalid command "get udp connection info". In this case, the individual command words are valid from the command word translation table 20, however, the sequence is invalid. In particular, the command word "get" having a token value of "3" reaches the token-command key pair 30b, however the command word "udp" having a token value of "7" does not reach any child of the tree element 24a. Hence, the parser 14 uses the last valid command key ("6") in step 54 based on the matching token for the first valid word located in the token-command key pair 30b. The command key is mapped to a selected one of the translators 16 in an attempt to provide a command to the corresponding resource 18. If the selected resource 18 determines that the command is invalid, the selected resource 18 at that time may prompt the user for a correct command.

The disclosed arrangement enables the use of generic commands for multiple OAM tools that have respective command syntax, resulting in a single point of entry for administering and maintaining complex software based systems. The disclosed arrangement provides the user a single set of commands and syntax to learn, facilitating the use of multiple administrative and maintenance tools.

While this invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not limited to the disclosed embodiments, but, on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.